INFORMATION TECHNOLOGY JOURNAL

Information Technology Journal 11 (10): 1341-1356, 2012
ISSN 1812-5638 / DOI: 10.3923/itj.2012.1341.1356

A Visual Management and Monitoring Tool for Cross-organization Emergency Response Workflows

Tingting Chen, Qingtian Zeng, Faming Lu, Jian Sun and Jie Zou
College of Information Science and Engineering, Shandong University of Science and Technology, Qingdao 266590, China

Abstract: In the cross-organization Emergency Response System (ERS), some tasks may depend on the coordination between various organizations in the disposal process of an accident. So monitoring and commanding the task-flow during the process is needed. In this study, firstly, we analyze the basic principles and define task-flow model in cross-organization emergency response system. Then we design and develop a visual tool to manage and monitor cross-organization workflow, tasks operations and cooperative control relationships among different organizations. The study introduces the architecture and the basic functions of the tool in detail and illustrates the feature technologies, such as the synchronization mechanism between different workflow editors, multi-view representations of workflow, workflow recommendation and dynamic monitoring over tasks' execution statuses, etc. Finally, the study shows the significance and value of the tool in the emergency disposal process of a coal mine accident instance.

Key words: Emergency response system, cross-organization, workflow, visualization, coordination

INTRODUCTION

Workflow is a conception which can bring computerized facilitation or automation for a business process. It is concerned with the automation of procedures in whole or part. In the procedures, documents, information or tasks are passed between participants according to a defined set of rules to achieve, or contribute to, an overall business goal (WfMC, 1996). Workflow management technology has been widely used in the emergency response management (Zhan, 2008). In the ERS, an emergency response process requires to issue and execute a set of tasks that have specified order of seniority. The task-flow (or workflow) could be represented and managed in different ways. Currently, workflow modeling techniques are relatively more mature. Of the worldwide workflow products, IBM's MQSeries workflow, Action Technologies, Inc. Metro, FileNet's Visual Workflow, Adobe's Adobe Workflow Server and PAVONE Espresso are the most famous. In addition to those commercial products, many universities and research institutions also contribute a lot in this area. For example: Exotica/FMQM (FlowMarkon Message Queue Manager) was developed by IBM's Almaden Research Center; Meteor (Managing end-to-end Operation) and WIDE (Workflow on Intelligent and Distributed Database Environment) were contributed by Organization of Computer Science University of Georgia. However, the workflow research has a relatively late start in China. One of the well-known researches is Qinghua University's CIMFlow, based on the research on CIMS application by Wu Cheng, Professor Fan Yushun and Professor Shi Meilin. They also accomplished the workflow management system based on Web and CORBA (Fan, 2001).

An overview of Workflow Management: From Process Modeling to Workflow Automation Infrastructure (Georgakopoulos *et al*., 1995) and ACTA: A framework for specifying and reasoning about transaction structure and behavior (Chrysanthis and Rammaritham, 1990) identify some relatively mature technologies and workflow modeling methods. Modeling and Parsing of Web Based Visualized Workflow and Research and Achieve on visualization tools of Workflow modeling (Li, 2003) describe several visualization workflow modeling methods. Shahzad and Rashid (2005) proposed architectural framework of workflow management system for heterogeneous and distributed environment which incorporates web-enabled independent interface for clients to execute workflows. Zhu *et al.* (2010) proposed a routing-based dynamic workflow and introduce agent technology into workflow management systems. Some workflow models are presented for workflow management system (Owaied *et al*., 2011; Li and Du, 2009). However, one emergency response

Corresponding Author: Qingtian Zeng, College of Information Science and Engineering, Shandong University of Science and Technology, Qingdao 266590, China

process often requires a number of organizations to participate. In this aspect, researches about cross-organization collaboration in emergency response includes making the use of Petri net theory to model the task and making up the collaboration model among organizations (Meng *et al*., 2011), modeling about emergency impact assessment, resource utility classification and mapping between impact and utility with a domain ontology (Sun *et al*., 2011). Some construction methods are proposed to obtain the process expressions of all kinds of Petri nets based on decomposition (Zeng, 2008), many analysis and algorithms for the modeling Petri Nets are also given and developed (Zeng, 2011; Cui *et al*., 2011; Du and Guo, 2009). Besides, Alfize (2002) studied the metrics evaluation for industrial OO Petri nets models. However, most workflow management tools are difficult to be directly applied to emergency response disposal process. On one hand, those tools only support workflow model under specific areas, lack of multi-task coordination mechanism which limits the representation power. On the other hand, they are imperfect in modeling and simulation. In this study, we create a visual management and monitoring tool for cross-organization emergency response process in ERS, monitoring task's actions, such as issued and executed and controlling tasks cooperatively among organizations. Representing task-flow in different views facilitates task coordination among organizations.

THE PRINCIPLE OF CROSS-ORGANIZATION ERS AND THE TASK-FLOW MODEL

In ERS, every emergency response process involves several organizations to work together to complete a series of tasks. These tasks may need either one organization to proceed independently, or multiple organizations to cooperate and communicate mutually.

The principle of cross-organization ERS: In a functional perspective, a typical cross-organization ERS consists of one command center and multiple emergency response organizations. The command center is responsible for creating emergency task-flow, issuing task commands to all organizations and monitoring the task's actions during the emergency response process. The organizations execute the tasks following instructions from the command center. During the execution, different organizations may be interactive about the task information. During or after task execution, each organization can feed back task statuses to the command center. Figure 1 shows the principle of cross-organization ERS. During emergency response process, the interactions and dependent relations among organizations can be divided into three categories:

- Some tasks of one organization must follow other tasks of another organization
- Several organizations participate in the same task
- There are message deliveries between organizations during the task's execution

The task-flow model of cross-organization ERS: This section presents the formal models of task and workflow in cross-organization emergency response process.

Definition 1: In cross-organization emergency response, each task is defined as a six-tuple:

Task = <Name, State, Organizations, MessagesReq, MessagesSent, PostTasks>, including:

- **Name:** The name of the task can be used for identification
- **State:** The current implementation status of the task

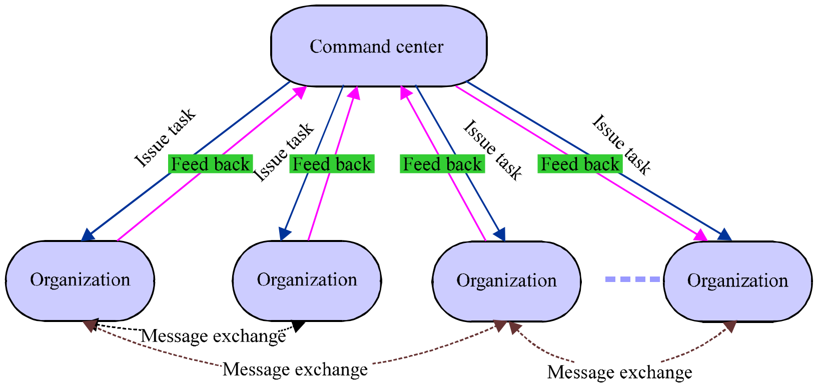

Fig. 1: The principle of cross-organization ERS

- **Organizations:** The set of the organizations which participate in the task. It should not be empty
- **MessagesReq:** The required message collection from other tasks during the execution, the collection can be empty
- **MessageSent:** The sent message collection to other tasks during the execution, the collection can be empty
- **PostTasks:** A collection of tasks following the current task. In other words, the tasks in PostTasks would be fired until the completion of the current task. When the set is empty, it means the current task is the last task

According to Definition 1, for a task t_1, the name of t_1 is denoted as t_1, so are the other properties. The number of elements in set S is denoted as |S|, e.g., the number of elements in Organizations of t_1 is indicated as $|t_1$. Organizations|. The state of t_1 has a single value from a finite set, i.e., t_1. State $\in$ {"Unissued", "Issued", "Signed", "Implementing", "Completion", "Revoked"}. And we constrain that t_1.Organizations $\neq \emptyset$.

Definition 1 implies three possible task coordination modes in organizations O_1 and O_2:

- The sequential mode: For two tasks t_1 and t_2, if $t_2 \in t_1$.PostTasks, t_1.Organization = $\{O_1\}$ and t_2.Organizations = $\{O_2\}$, then t_1 and t_2 is in sequential mode
- The task-share mode: For a task t_1, if there are two organizations O_1 and O_2 satisfying $O_1 \in t_1$.Organizations and $O_2 \in t_1$.Organizations, then t_1 is the shared task of O_1 and O_2
- The message-exchange mode: For two tasks t_1 and t_2, if t_1.MessagesReq$\cap t_2$.MessagesSent $\neq \emptyset$, or t_2. MessagesReq$\cap t_1$.MessagesSent $\neq \emptyset$ and $O_1 \in t_1$.Organization and $O_2 \in t_2$.Organization, then t_1 and t_2 is in the message- exchange mode

Definition 2: An emergency response process can be represented as a directed acyclic graph, denoted as a two-tuple ⟨V(process), E(process)⟩. Specifically, V (process) is the node set representing tasks, denoted as V(process) = $\{t_1, t_2 \ldots t_i \ldots\}$ in which t_i is a task as defined in Definition 1. E(process) = $\{\langle t_i, t_j\rangle \ldots \langle t_m, t_n\rangle \ldots\}$ ($\{t_i, t_j, t_m, t_n \ldots\} \subseteq$V(process)) is the arc set representing the binary relationship between tasks.

By Definition 2, one emergency response process can be modeled as a directed acyclic graph. The sequence relationship between tasks can be derived from task's PostTasks property. In the model, if there is a directed path from the vertex i to vertex j, then the vertex i is the precursor of j; j is the successor of i. If <i, j> is an arc in graph, i is the direct precursor of j; j is the direct successor of i.

In particular, the start node and end node in workflow of emergency response are defined as t_0 = ⟨'Root','',∅, ∅, ∅, postT⟩ and t_{end} = ⟨'End','', ∅, ∅, ∅, ∅⟩, where, t_0.postTasks = postT is a set of t_i which satisfies the following conditions: $t_i \in$V(process) and $\forall t_j \in$V(process), t_j.PostTasks. In addition, in the workflow of emergency response, if $\exists t_k \in$V(process, t_k.PostTasks = ∅), then make t_k.PostTasks = t_{end}.

For example, Table 1 shows a process of cross-organization emergency response.

We use a model similar to Fig. 2 to model an emergency response task-flow, where, the

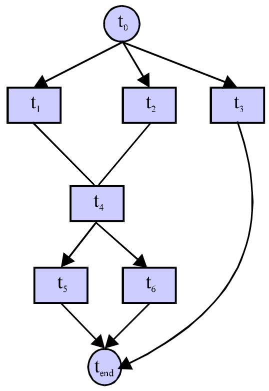

Fig. 2: An emergency response task-flow

Table 1: A process of cross-organization emergency response

	Name	State	Organizations	MessagesReq	MessagesSent	PostTasks
t_1	N_1	"Unissued"	{"Organization$_1$", "Organization$_2$"}	{Message$_1$}	¢	$\{t_4, t_5, t_6\}$
t_2	N_2	"Unissued"	{"Organization$_1$", "Organization$_2$", "Organization$_3$"}	¢	{Message$_2$}	$\{t_4, t_5, t_6\}$
t_3	N_3	"Unissued"	{"Organization$_1$", "Organization$_3$"}	¢	{Message$_1$, Message$_3$}	$\{t_{end}\}$
t_4	N_4	"Unissued"	{"Organization$_1$"}	{Message$_2$}	¢	$\{t_5, t_6\}$
t_5	N_5	"Unissued"	{"Organization$_3$"}	¢	¢	$\{t_{end}\}$
t_6	N_6	"Unissued"	{"Organization$_1$", "Organization$_2$"}	{Message$_3$}	¢	$\{t_{end}\}$

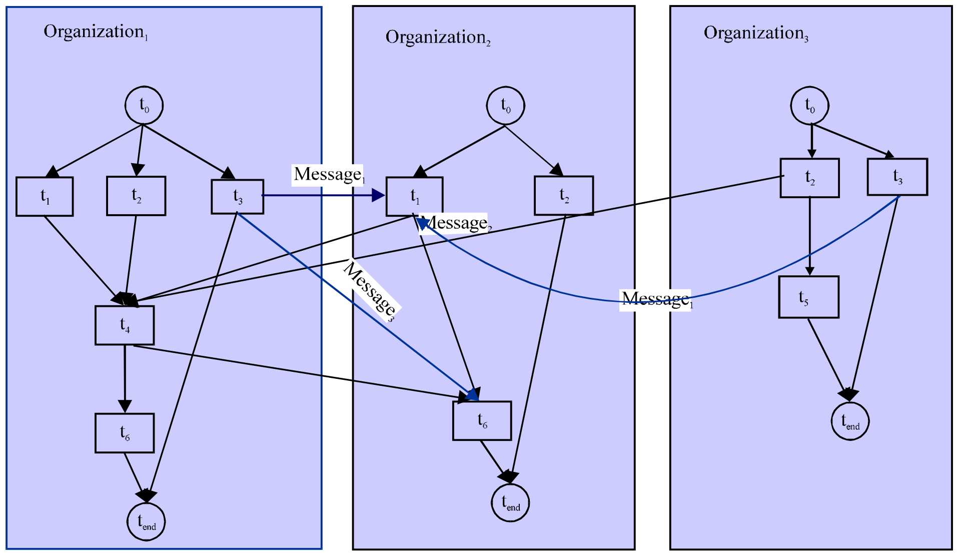

Fig. 3: Task-flow of cross-organization

rectangle node in Fig. 2 represents a specific task, start node and end node are presented with two circles, respectively.

As shown in Fig. 2, the tasks which are issued by command center and their relationships form a task-flow model, can be expressed as process = $\langle V, E\rangle$, where, V = $\{t_0, t_1, t_2, t_3, t_4, t_5, t_6, t_{end}\}$, E = $\{<t_0, t_1>,<t_0, t_2>, <t_0, t_3>, <t_1, t_4>, <t_2, t_4>, <t_4, t_5>, <t_4, t_6>, <t_3, t_{end}>, <t_5, t_{end}>, t_6, t_{end}\}$.

Figure 3 shows the task models formed in the organizations when carrying out tasks, respectively after receiving the mission from the command center. A rectangular region is called a swimlane meaning one organization scope. This term will be used directly in the following sections.

THE ARCHITECTURE OF VISUAL MANAGEMENT AND MONITORING TOOLS AND FUNCTIONS

Architecture: In order to monitor the execution of task for cross-organization emergency response intuitively and clearly in ERS, we construct a visual management and monitoring tool. Figure 3 provides a logic description of its structure. In this tool, the integrity of a particular emergency response process can be solved as atypical task-flow, because of the series of standardized task sets if formed. Therefore, task-flow library is a set which stored some examples of the emergency response process, all these typical task-flows have its own type and level.

The overall structure is shown in Fig. 4, a visual management and monitoring tool consists of the two major components, i.e., the command center monitoring task-flow and the organizations monitoring the task-flow.

The part of the command center monitoring task-flow is the visual environment after center commanders logged in the system, including the task-form editor and task-flow editor. The task-form editor displays the tasks included in the emergency response process in table format, for the sake of supervision and adjustments. The task-flow editor is divided into several sub-environments and is used to monitor and schedule the current emergency response process. When monitoring task-flow in emergency response process, we can refer to typical task-flows stored in the task-flow library, visualize them and make adjustments to the current situation.

The part of the organizations monitoring task-flow is a visual environment after organizations staff logged in the system. In this environment, each organization could do some operations on tasks issued from the command center, or exchange message with other organizations that have task-flow relationships, or feed back related information to the command center.

The command center monitoring task flow: The environment of command center monitoring task-flow includes the task-form editor and task-flow editor. The following will introduce the functions about the two editing environment.

In the task-form editor, emergency response tasks are listed in a table. Buttons on the table are associated with

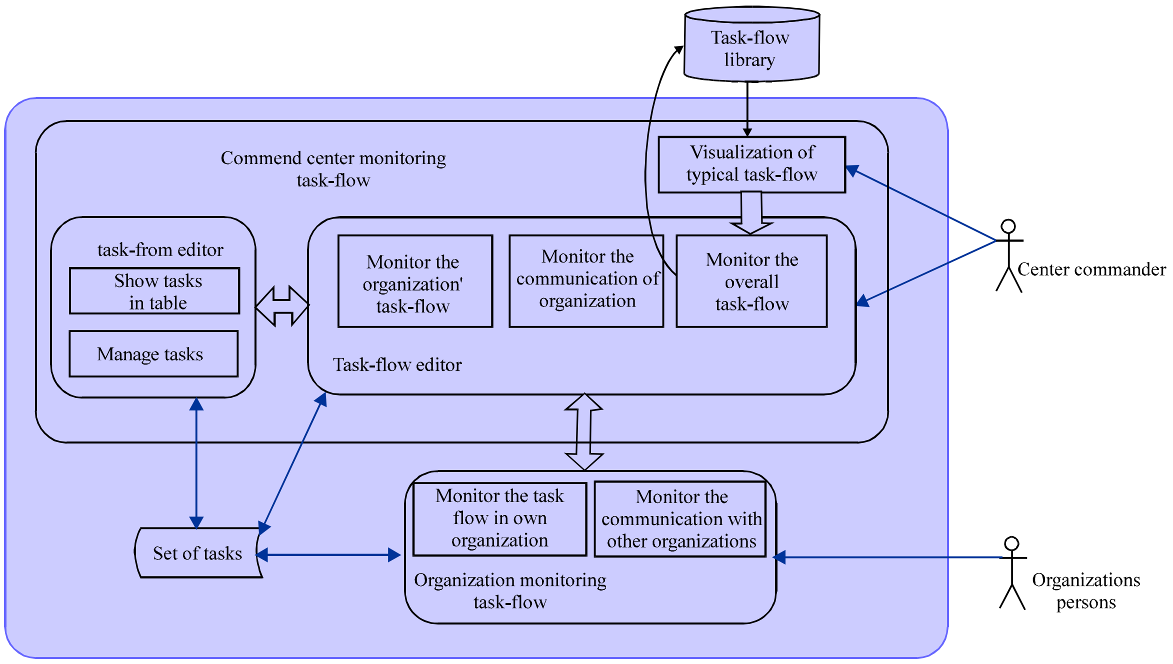
Task-flow library
Commend center monitoring task-flow
Visualization of typical task-flow
task-from editor
Show tasks in table
Manage tasks
Monitor the organization' task-flow
Monitor the communication of organization
Monitor the overall task-flow
Task-flow editor
Center commander
Set of tasks
Monitor the task flow in own organization
Monitor the communication with other organizations
Organization monitoring task-flow
Organizations persons

Fig. 4: The structure of visual management and monitoring tool

different operations, including task inquiry, creation, edition, assignment, withdrawal, completion and so on. Center commanders can quickly define the properties of new tasks, add them to the table; also can carry on related operations to selected tasks using the appropriate action buttons on the table. Finally, all operations are showed in the emergency response task-flow.

The task-flow editor can be divided into four parts: monitoring the overall task-flow, the command center monitoring the task-flow of organizations, monitoring messages exchanging among organizations and the visualization of typical task-flow in task-flow library.

Monitoring the overall task-flow: In this part, the command center manages and coordinates the overall task-flow rather than tasks distributed in specific organizations. The operation on the task node will synchronously affect all the participating organizations' task-flow in this emergency response. For those "Unissued" tasks, we can view details in the task-flow, add a follow-up tasks, edit, issue, revoke; while for those issued ones, we can only do the operations of viewing details, adding follow-up tasks and removing the task.

The command center monitoring the task-flow of organizations: In this part, center commanders can monitor the task-flows of different organizations. The operations include:

- **Task detail:** View the details information of selected task
- **Add task:** Add a new task, take the selected task node as a precursor task
- **Edit task:** Edit the selected task, only be available for the task nodes whose property of state is "Unissued"
- **Assign task:** Issue the selected task to the relevant organizations, only be available for the task nodes whose property of state is "Unissued"
- **Revoke task:** Revoke the selected task node
- **Attention:** The color of the task nodes vary from the task states in the view. Different node colors indicate different task states

Monitoring messages exchanging among organizations: Throughout the emergency response process, each organization executes its task-flow independently. In other words, the statuses of internal tasks within one organization can be shielded in the emergency response, only remaining interaction information such as coordination messages and task dependent relations between different organizations. The shielding mechanism both ensures the privacy of the task-flow within the organizations and simplifies the coordination of the relationship among organizations.

For example: in Fig. 5, the task t_4 in organization$_1$ is following task t_1 and task t_2, t_1 has completed by organization$_1$ and organization$_2$ together, so only the task t_1 in organization$_2$ is displayed according to the shielding

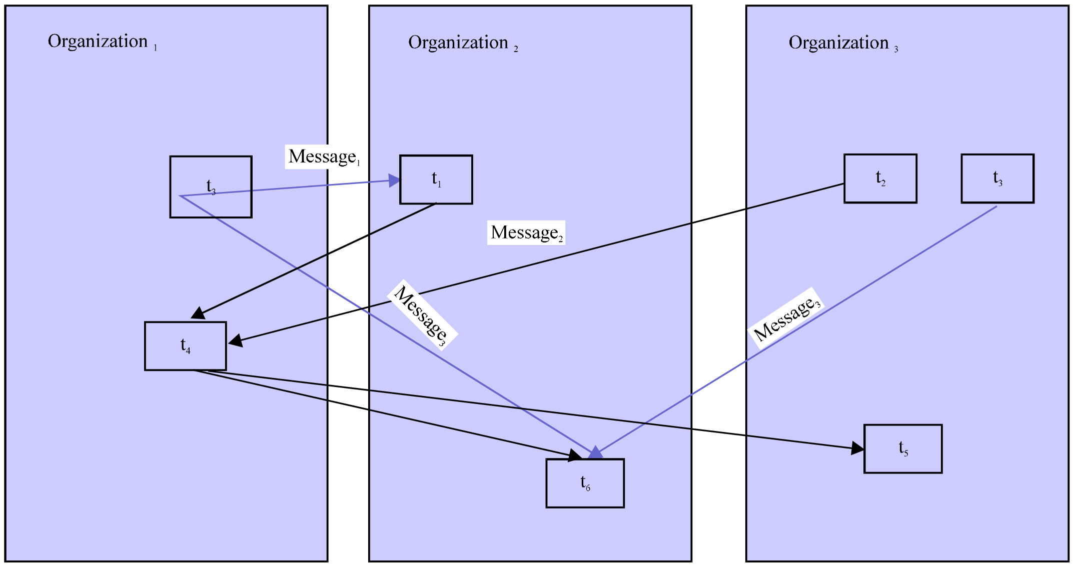

Fig. 5: Monitoring coordinate relationship among organizations

mechanism. The value of property MessageSent set is {Message$_1$, Message$_3$} in t$_3$, the value of property MessageReq set is {Message$_1$} in t$_1$, the value of property MessageReq set is {Message$_3$} in t$_6$. Figure 5 also shows the message exchange of tasks among organizations. So, it is not only to ensure the independence and security operations within the various organizations but also monitor the link among organizations in a whole perspective.

The visualization of typical task-flow in task-flow library: People in command center can extract typical task-flows from task-flow library, visualize them and monitor the emergency response task-flow based on the typical task-flow. When importing typical task-flows, the system can recommend the best match task-flows according to the type and level of the current incident. After center commanders select one typical task-flow, the system will load the related follow-up tasks based on the existing task-flow structure.

The operations of management and monitoring include:

- **Copy a single task:** Select a task node of the typical task-flow, copy it to emergency response task-flow, under a certain task node which is the precursor task of the coped task node
- **Copy task-flow fragment:** In the typical task-flow, select the task-flow fragment (some task nodes with order) you want to add to the emergency response task-flow, perform the operation of "copy task-flow fragment" and then copy it in the emergency response task-flow, under some one node, task-flow fragment
- **Add precursor relationship:** Select two task nodes in the emergency response task-flow, perform “add precursor mission” operation, you can set the task before and after relationship between the two selected nodes. The precursor relationship can not be repeatable set. If there were cycles in task-flow after adding precursor relationship, then the relationship can’t be created, too
- **Remove the precursor relationship:** Select two task nodes with the sequence relationship in emergency response task-flow, perform “remove the precursor relationship” operation and you can delete the sequence relationship between two nodes. If the task has no predecessor tasks after the operation, this task will be automatically added under the root node in the task-flow
- **Task details:** View the details information of selected task
- **Add task:** Add a new task, take the selected task node as a precursor task
- **Edit task:** Edit the information of selected task
- **Remove task:** Remove the selected task and its follow-up tasks

The organizations monitoring task-flow: The functions of this part consist of two parts: The first one is that, the person in organization can perform some operations for the task-flow in this organization listed in the next

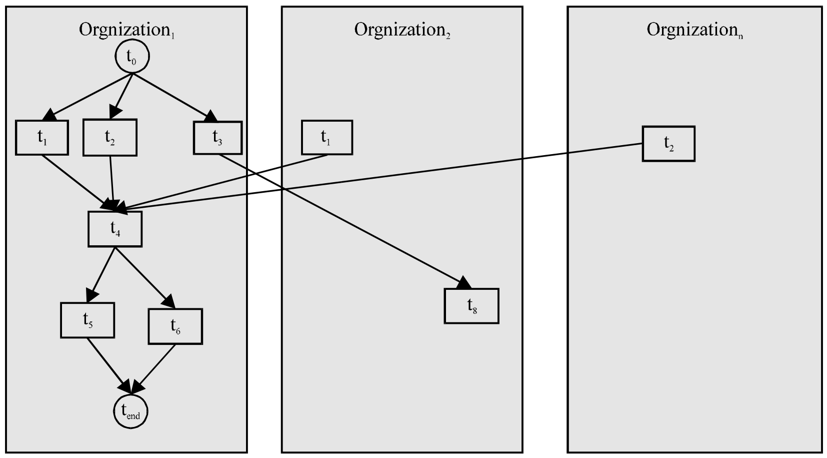

Fig. 6: Visual environment for organizations staff

paragraph; the second one is that, monitoring the task interacting with other organizations. View of the structure shown in Fig. 6. The monitoring operations in task-flow for organizations including:

- **Task detail:** View the detail information of the selected task
- **Sign-up task:** This option is only available for the task nodes whose property of state is "Issued". Select "Sign-up", fill in the dialog with the name, indicating receipt of the task. Then organization could proceed with the task
- **Start task:** This option is only available for the task nodes that do not start after assigned. Select this option to fill in the dialog with his/her name, then you can start the selected task
- **Feedback task:** Transmit the message of task situation to command center. Select the option, the feedback window popped up, completing people's information and the feedback content, judging whether it is urgent. Feedback person in the window is the one landed the system by fault. After completing the information, it can be fed back to the command center
- **Application for completion task:** This is available only for the executing task. Select the option, a window popped up, fill with the applicant's name (the one landed the system by fault) and finish feedback information. After that, it can be transmitted to the command center, and then it is up to center commanders whether end this task

As shown in Fig. 6, in the part of organizations monitoring task-flow, except for monitoring the task-flow in one's own organization, people can view the communication with other organizations by some tasks. In the task-flow, if there are some tasks' predecessor tasks or successor tasks in other organizations, people can view the detail information of predecessor tasks or successor tasks but can not edit them, such as t_1 and t_8 in organization$_2$. It is more convenient for people in one organization to control the task-flow in their own organization, when they can realize the communication with the tasks in other organizations.

TECHNICAL FEATURES OF VISUAL MANAGEMENT AND MONITORING TOOL

The visual management and monitoring tool employs several featured technologies including synchronization between different task editors, multi-view representation, task-flow recommendation and dynamic monitor.

The synchronization between task-form editor and task-flow editor: The task-form editor and task-flow editor in command center monitoring the task-flow are synchronous. The synchronization can be achieved from the three levels of view-logical control-model. The view level consists of task-form editing environment and task-flow editing environment. The logic control level is used to control show effects of both environments. The model level is mapping the task node model of emergency response. For example, the center commander adds a new task in the task-form editor, the following steps procedure could achieve the synchronization with task-flow editor:

Step 1: The user adds a new task in the task-form editing environment, gives the name, the participant organizations and the successor tasks of the new task, respectively

Step 2: The new task information is transmitted to logic control part of task-form, the name, the participant organizations, and the successor tasks are defined as $name_i$, $organizations_i$, $posttasks_i$

Step 3: To build a new task instance t_i, the properties are set as $state_i$ = 'not issued' and $MessagesReq_i$ = $MessagesSent_i = \varnothing$ by default, that is ti = $\langle name_i$, 'not issues', $organizations_i$, $\varnothing$, $\varnothing$, $posttasks_i\rangle$. To set a flag $changeflag_i$ = true for t_i

Step 4: In the logic control part of task-flow, to read all task instances and their flags, if the flag of the task instance t_k is $changeflag_k$ = true, then display the task node with a particular color in the task-flow editing environment

Similarly, if the center commander modify some task's information in the task-flow editing environment, the changes also can pass by the logic control part of task-flow, task node model and the logic control part of task-form, finally appear in the task-form editing environment.

The synchronization of task-form editor and task-flow editor is shown in Fig. 7.

The multi-view visualization of task-flow: In Fig. 8, multi-view visualization is fulfilled in two ways: on one hand, according to the roles of the people logged in ERS, the system presents different views; on the other hand, it represents different views based on the different requirement of center commander.

The visual environment of the tool is set to two different parts in accordance with the roles of landing people, one is the task-flow monitoring environment used by center commanders, the other is used by organizations staff. During the user's login process, the system will verify the user's role and assign corresponding authority according the account number. Center commander will be directed into the command center monitor environment, while organizations staff will be directed into organization monitor environment.

After center commander logged in, he can monitor and manage the task-flow in four kinds of views, i.e., the overall view of task-flow, the organizational view of task-flow monitored by center commander, the communication view of cross-organization and the view from typical task-flow in library.

The overall view of task-flow is displayed as follows:

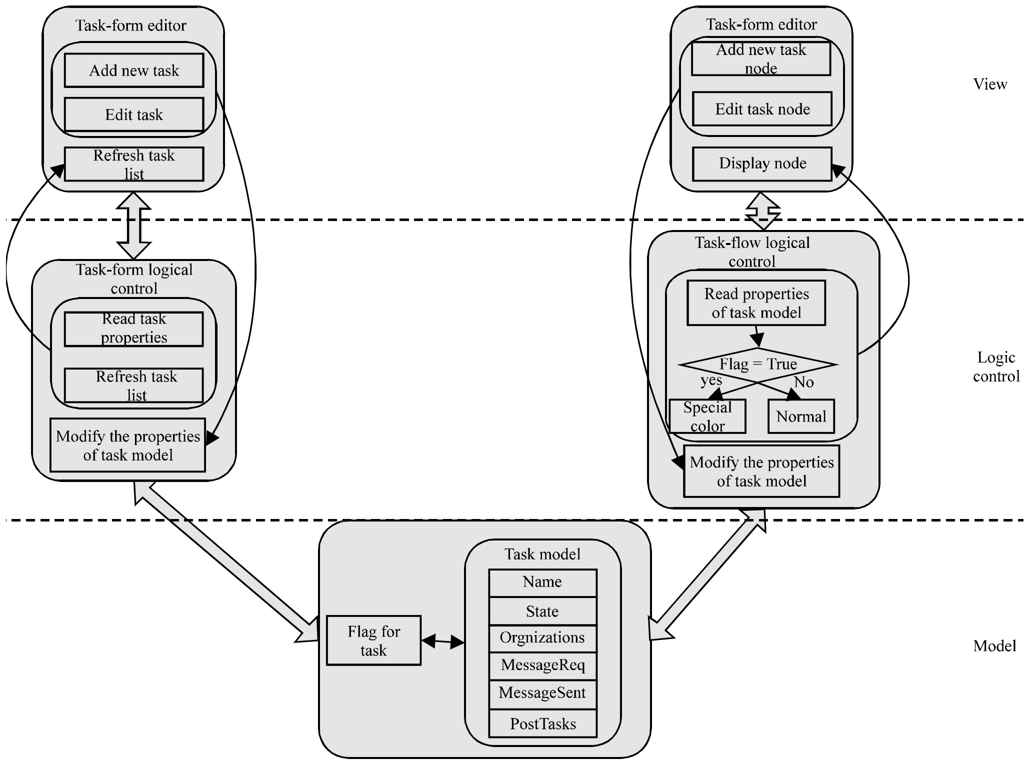

Fig. 7: Synchronization of task-form editor and task-flow editor

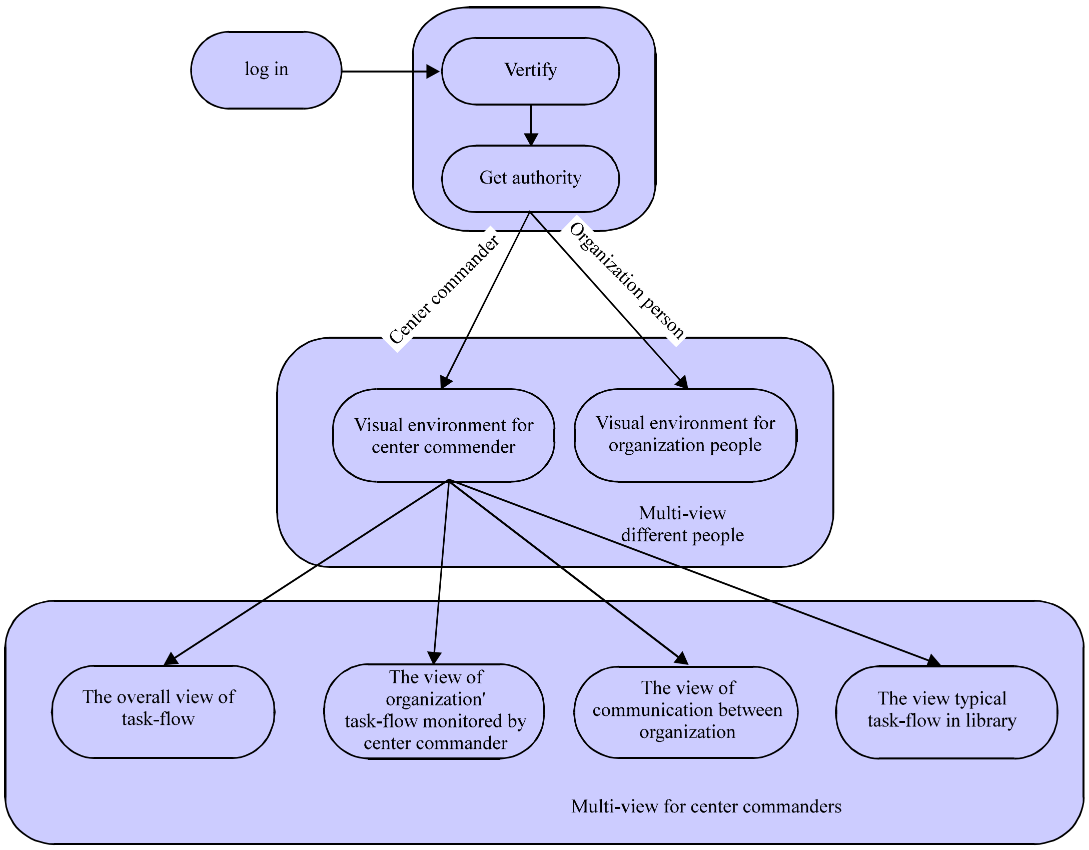
log in
Vertify
Get authority
Center commander
Organization person
Visual environment for center commender
Visual environment for organization people
Multi-view different people
The overall view of task-flow
The view of organization' task-flow monitored by center commander
The view of communication between organization
The view typical task-flow in library
Multi-view for center commanders

Fig. 8: Multi-view visualization of task-flow

Step 1: Read the task set of emergency response, denoted as taskList = {t_1, t_2, ...,t_n}, initialize the start node t_0 = ⟨'Root', '', ∅, ∅, ∅, ∅⟩ and the end node t_{end} = ⟨'End', '', ∅, ∅, ∅, ∅⟩

Step 2: Take t_i∈ taskList ($0<i\leq n$) in turn, according to the information of task t_i to establish a task node, if the node is not in view, then turn to take t_p∈t_i.PostTasks ($0<p\leq|t_i$.PostTasks$|$), if there is no corresponding task node of t_p in the view, add it and add a directed arc ⟨t_j, t_p⟩

Step 3: Take t_j∈taskList ($0<i\leq j$) in turn, determine whether there is t_{pre}∈taskList ($0<pre\leq n$) to make ⟨t_{pre}, t_i⟩ appearing and determine whether there is t_{post}∈taskList ($0<post\leq n$) to make ⟨t_j, t_{post}⟩ appearing. If t_{pre} is not existed, add a directed arc ⟨t_0, t_j⟩; if t_{post} is not existed, add a directed arc ⟨t_j, t_{end}⟩

- The organizational view of task-flow monitored by center commander is displayed as follows: First, get the set of all organizations involved in emergency response, remove reputation, display the view in swimlane style; then, analyze all tasks' organizations property, find the task in which swimlane, add the task nodes to their swimlanes; next, analyze the tasks' PostTasks property, find the relationship among them and add the connection; finally, add the start node and end node in each organization's swimlane, all nodes without predecessor tasks are connected to the start node, all nodes with no successor tasks are connected to the end node. Eventually, the organizational view of task-flow monitored by center commander is shown.

The process used the algorithm:

Algorithm 1:

Step 1: Get the task set of emergency response, denoted as taskList = {t_1, t_2,..., t_n}, initialize the set of organizations orgList = ∅, i, j is the iterator of taskList, k is the iterator of orgList and set that i = 1, j = 1, k = 1

Step 2: Take t_i∈taskList, if t_i.Organizations ⊄ orgList, then orgList = orgList ∪ t_i.Organization, i = i+1

Step 3: If i>n, go to Step 4; otherwise, go to Step 2

Step 4: Take $org_k \epsilon orgList$, draw the swimlane in the view, and named org_k. k = k+1

Step 5: If i>|orgList|, go to Step 6; otherwise, go to Step 4

Step 6: Take $t_j \epsilon taskList$, according to the task information to create a task node

Step 7: Take $O_m \epsilon t_j.Organization(0<m \leq |t_j.organizations|)$ in turn, if $O_m = org_h(0<h \leq |orgList|)$, add the task node in org_h

Step 8: Take $t_p \epsilon t_j.PostTasks(0<p \leq |t_j PostTasks|)$, if the task node of t_p has been in the swimlane, then abandon it; else, create a task node according to t_p, take $O_a \epsilon t_p$. Organizations $(0<a \leq |t_p$. Organizations$|)$ in turn, if $O_a = org_h$ $(0<h \leq |orgList|)$, then add the task node in the swimlane org_h and add a directed arc $\langle t_j, t_p \rangle$

Step 9: If j<n, go to the end; otherwise, j = j+1, go to Step 6

- In the cross-organization communication view, when adding a task node in swimlane, we need to not only analyze the dependent relations of tasks but also compare with the swimlane of the successor task nodes. If the successor task nodes are in the same swimlane with the current task node, then abandon them; Otherwise add the current task node in its swimlane, add the successor task nodes which are not in the same swimlane in their own swimlanes and add the connection between them. The above process combines the Step 7 and Step 8 in Algorithm 1 into the following ones:

Step 1: Take $O_m \epsilon t_j.Organization(0<m \leq |t_j.Organizations|)$ in turn, p is the iterator of t_j.PostTasks and set p = 1.

Step 2: Take $t_p \epsilon t_j$.PostTasks, if the task node of tp already exists in the swimlane, then abandon it; otherwise, take $O_a \epsilon t_p.Organization$ $(0<a \leq |t_p.Organization|)$ in turn, if $O_a = O_m$, abandon tp, if $O_a \neq O_m$, create the task node of t_p in the swimlane O_a and add a directed arc $\langle t_j, t_p \rangle$.

Step 3: If $p>|t_i.PostTasks|$, go to Step 9 in Algorithm 1; otherwise, p = p+1, go to Step 2.

Thus, the final view shields the task-flow within the organizations, leaving only the task nodes and relations among different organizations.

- The view from typical task-flow in library is displayed as follows: First, get all task-flows from the task-flow library to form a list. Then extract the properties of one task-flow from the list to from a typical task-flow model; at the same time, get the properties of all tasks in emergency response to from an emergency response task-flow model. Finally, the typical task-flow model and the emergency response task-flow model are displayed in the view together, center commander can editor and monitor the emergency response task-flow through the view operation button, on the basis of the typical task-flow

Realization of the process is shown in Fig. 9.

The recommendation of task-flow: When center commander manages and monitors the emergency response task-flow by using the typical task-flow from the task-flow library, the system can recommend some typical task-flows based on the type and level of the current emergency incidents. After the center commander chooses one typical task-flow, the system can recommend some follow-up tasks based on the current situation of emergency response task-flow. Recommended methods are shown in Fig. 10, including typical task-flow recommendation and follow-up tasks recommendation.

- Typical task-flow recommendation: In ERS, the current emergency incident has specific type and level, the typical task-flows in task-flow library also have their own type and level. First, extract the type and level of current emergency incident; then find the typical task-flows which have the same type and level with the current emergency incident; finally, the recommended typical task-flows are showed in the view as a tree list. The realization of the process is shown in Fig. 11
- Follow-up tasks recommendation: After center commanders selecting one typical task-flow, they can see the typical task-flow and the emergency response task-flow together. According to the comparison of two task models, the recommended follow-up tasks are shown in the typical task-flow

Specific algorithm "Recommend (taskFlow, schemeFlow)" is as follows:

Algorithm 2:
Input: The task set of current emergency response taskFlow = $\{t_1, t_2,...,t_m\}$, the task set of selected typical task-flow schemeFlow = $\{t_1, t_2,...,t_n\}$.

Output: The recommended task set recomList from schemeFlow.

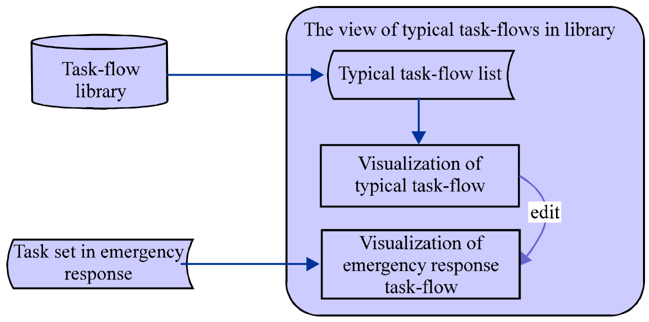
Task-flow library
The view of typical task-flows in library
Typical task-flow list
Visualization of typical task-flow
edit
Task set in emergency response
Visualization of emergency response task-flow

Fig. 9: The view from typical task-flow in library

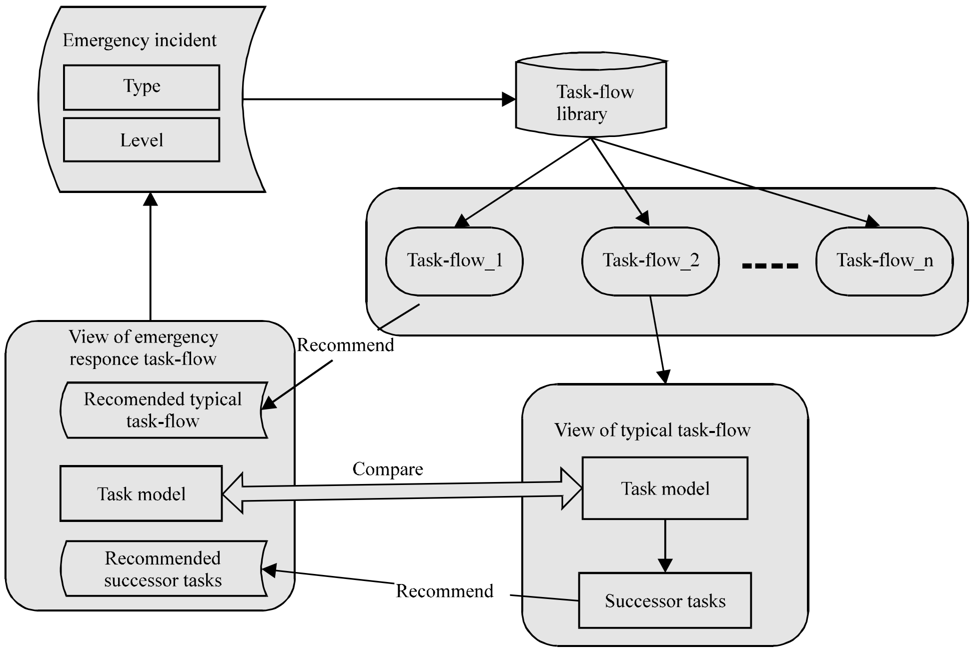

Fig. 10: Recommendation of task-flow

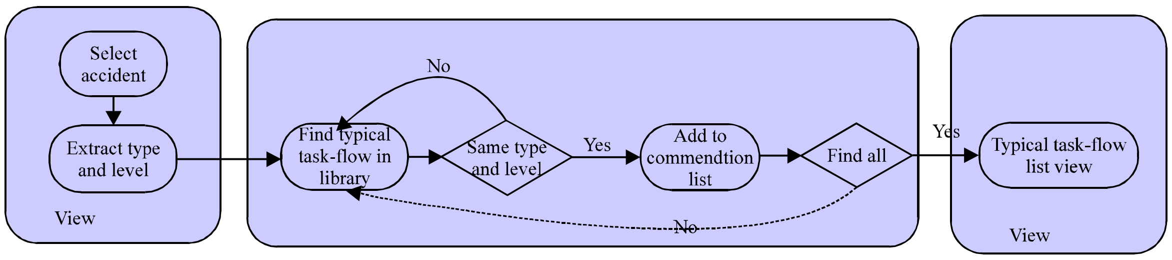

Fig. 11: Process of typical task-flow recommendation

Step 1: Let i and j are iterators of taskFlow and schemeFlow, initialize that i = 1, j = 1 and initialize a set of recommended tasks recomList = ∅.

Step 2: Take t_i∈taskFlow.

Step 3: Take T_j∈schemeFlow.

Step 4: If t_i.Name = T_j.Name, go to Step5; otherwise, j = j+1, go to Step 6.

Step 5: RecomList = RecomList∪T_j.PostTasks. j = j+1.

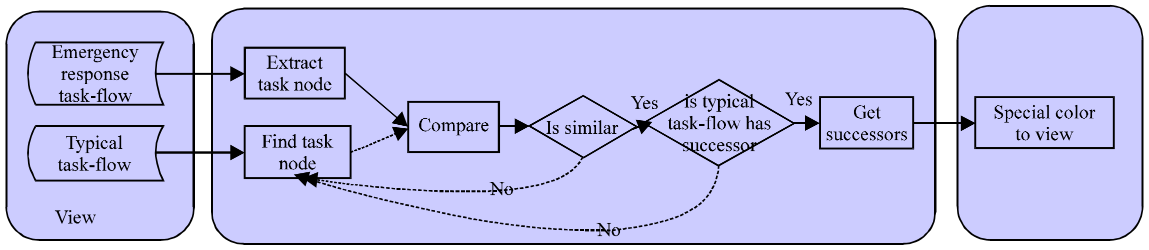

Fig. 12: Process of follow-up tasks recommendation

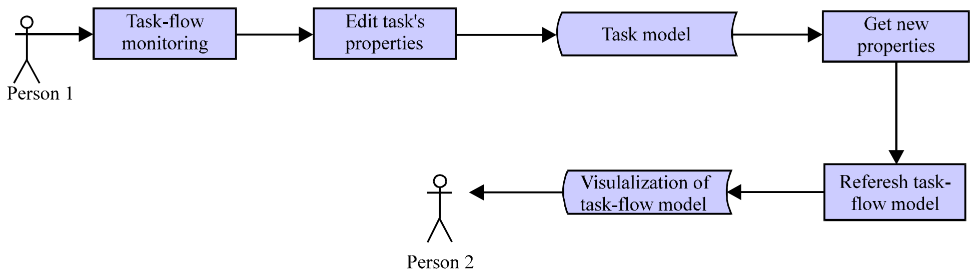

Fig. 13: Dynamically monitoring of task status

Step 6: If j>n, then i = i+1, go to Step 7; otherwise, go to Step 3

Step 7: If i>m, taking $t_r \in$recomList in turn, display t_r in the view with a particular color, the end; otherwise go to Step 2

The process of follow-up tasks recommendation is shown in Fig. 12.

Dynamic monitoring of task execution status: During the emergency response process, the statuses of node tasks in the task-flow follow the change of organizations. Under the view of the visual management and monitoring tool, people can monitor the change of task execution status dynamically.

The initial status of tasks created by the center commander is “Unissued”. The task status will become to “Issued” or “Revoked” if tasks are issued or revoked. Only the issued tasks can be operated by the organizations in the visual management and monitoring environment, the task state will change to “signed” or “implementing” after the task sign-up or start operation. If the command center approves all the applications to apply for completion of all the related organizations, the task states will become to “completion”.

The visual environment will update the monitoring page dynamically, that is to reread each property of emergency response tasks from time to time and rebuild task-flow model in the view. Under the visual management and monitoring environment, in order to let users monitor the change of the view process, the task node model will show different colors according to the difference of the property “state”. The realization is shown in Fig. 13.

In addition to the information of the task status, the change of other property information of tasks can be reread and refreshed in the current visual environment.

Therefore, after the operation for emergency response tasks, the people in organizations can monitor the task status and the change of each property in the visual environment without re-login. If the task status changes because of the monitoring operation, the center commanders can get the task information in time in the current visual environment by refresh from time to time.

AN INSTANCE UNDER THE VISUAL MANAGEMENT AND MONITORING TOOL

Taking the rescuing process of a coal mine accident in ERS as an example, we use the visual management and monitoring tool to monitor the process of this emergency response.

Once the incident named “Gas Explosion in No.1 Mine” is reported to ERS, the center commander will issue a series of sequential tasks in task-form editor or task-flow editor. The specific tasks are shown as in Table 2.

When the center commander enters the visual environment, they can operate the tasks in four views: The overall view of task-flow, the organizational view of task-flow monitored by center commander, the

Table 2: The task list of "Gas Explosion in No.1 Mine"

	Name	State	Organizations	MessagesReq	MessagesSent	PostTasks
t_1	"save people"	"Completion"	{"Security Section", "Production Section"}	¢	{"There is still fire"}	{ t_4, t_5, t_6}
t_2	"coordinate"	"Implementing"	{"Scheduling Office"}	¢	{"someone dead"}	{t_{end}}
t_3	"personnel evacuate"	"Signed"	{"Security Section", "Production Section"}	¢	{"There is still fire"}	{ t_5, t_6}
t_4	"control explosive source"	"Issued"	{"Production Section", "Technology Section"}	¢	¢	{ t_7}
t_5	"fire control"	"Issued"	{"Ventilation Section", "Production Section"}	{"There is still fire"}	¢	{t_{end}}
t_6	"air"	"Issued"	{"Ventilation Section"}	¢	¢	{t_{end}}
t_7	"site clearing"	"Issued"	"Production Section"	¢	¢	{t_{end}}
t_8	"assessment of loss"	"Unissued"	{"Finance Section", "Production Section"}	¢	¢	{t_{end}}
t_9	"pacify the wonnded"	"Unissued"	{"Production Section"}	{"someone dead"}	¢	{ t_8}
t_{10}	"investigating causes"	"Unissued"	{"Technology Section", "Production Section"}	¢	¢	{t_{end}}

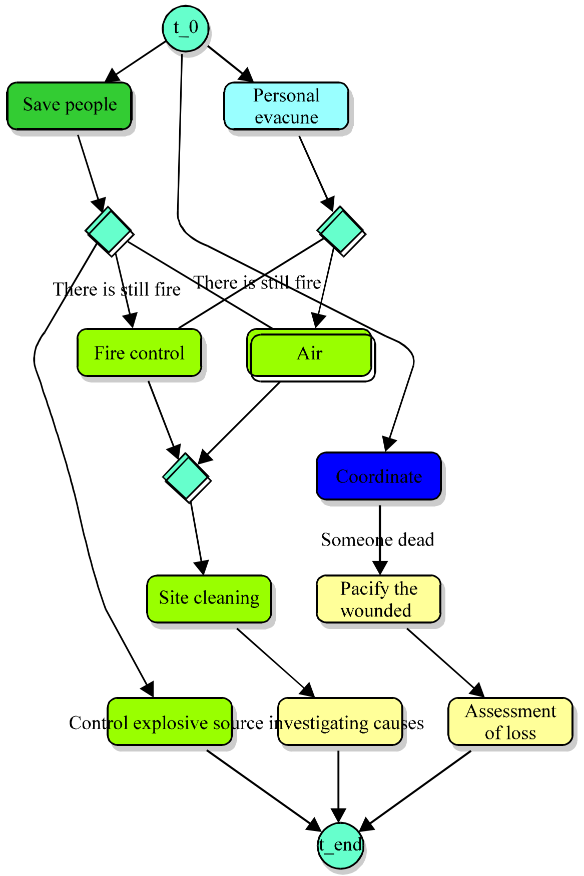

Fig. 14: Monitoring the overall task-flow

communication view of cross-organization and the view from typical task-flow in library. They can do operations such as task creation, deletion, edition, assignment and feedback.

The interface used for the command center to monitor issued tasks is shown in Fig. 14 and the color of node implies the task state; the interface used for the command center to monitor inner task-flow of one organization is shown in Fig. 15 and 16, displays the relations of tasks among monitoring organizations without concerning inner task-flow process.

Figure 17 shows the interface that the command center imports the typical task-flow of "Gas Explosion" from the task-flow library and visualizes it. The center

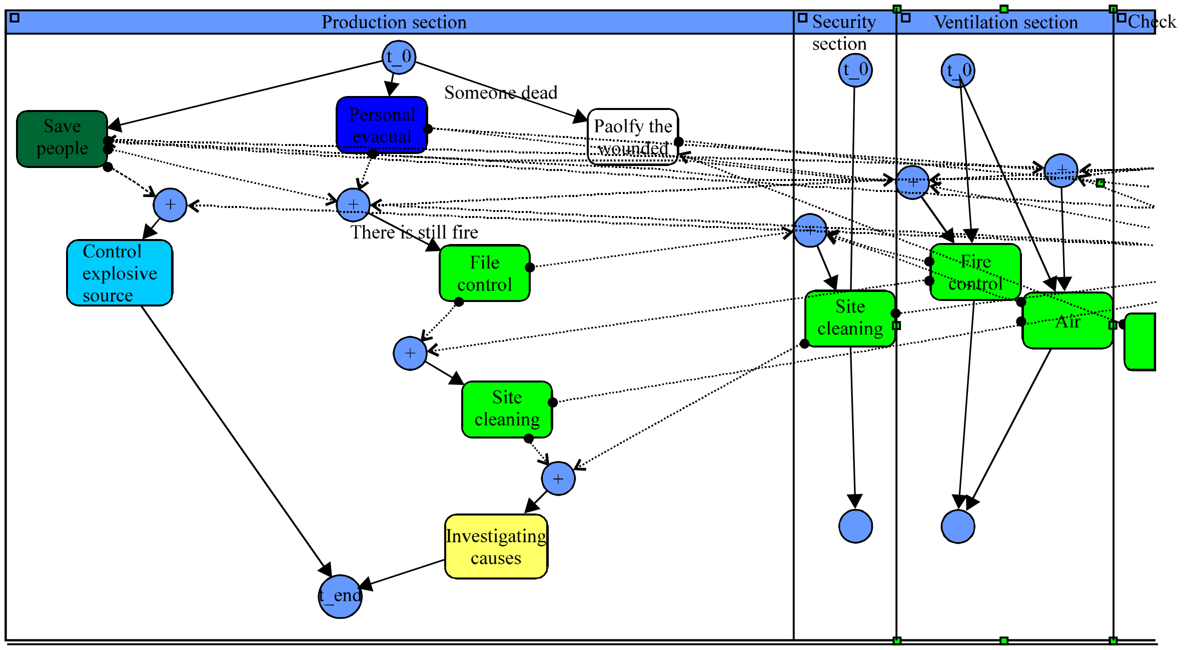

Fig. 15: Command center monitoring the task-flows in organizations

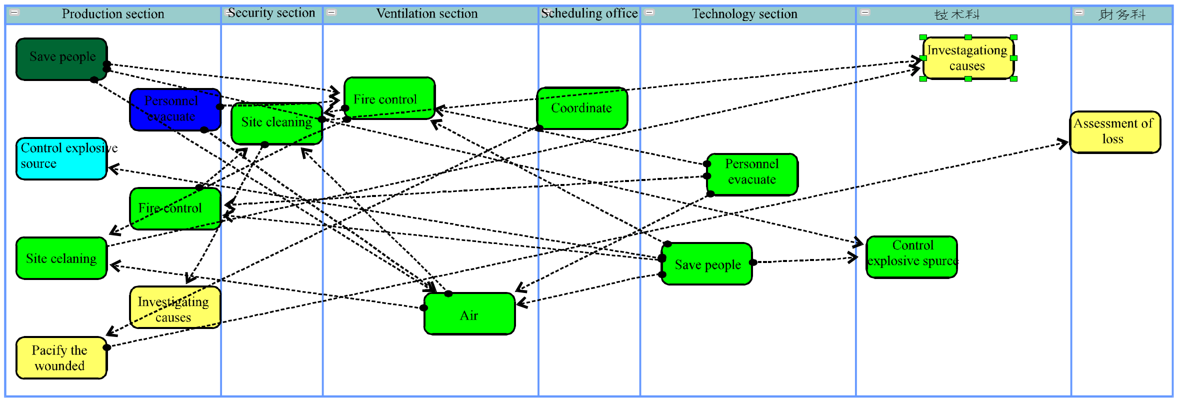

Fig. 16: Monitoring the communication among organizations

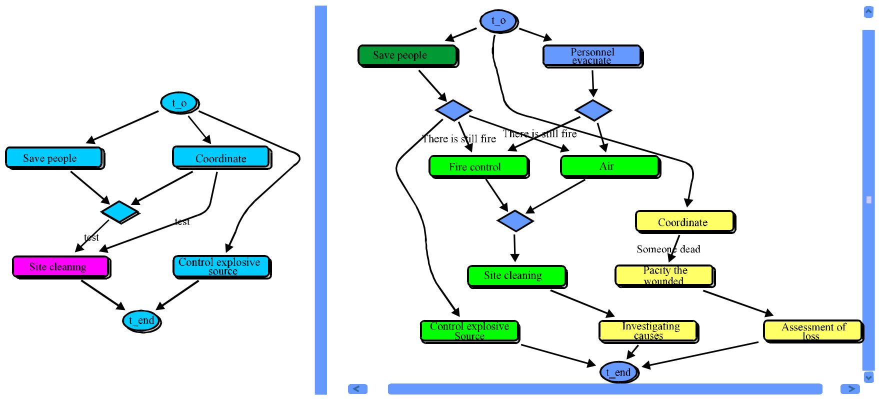

Fig. 17: Editing task-flow in emergency response by typical task-flow

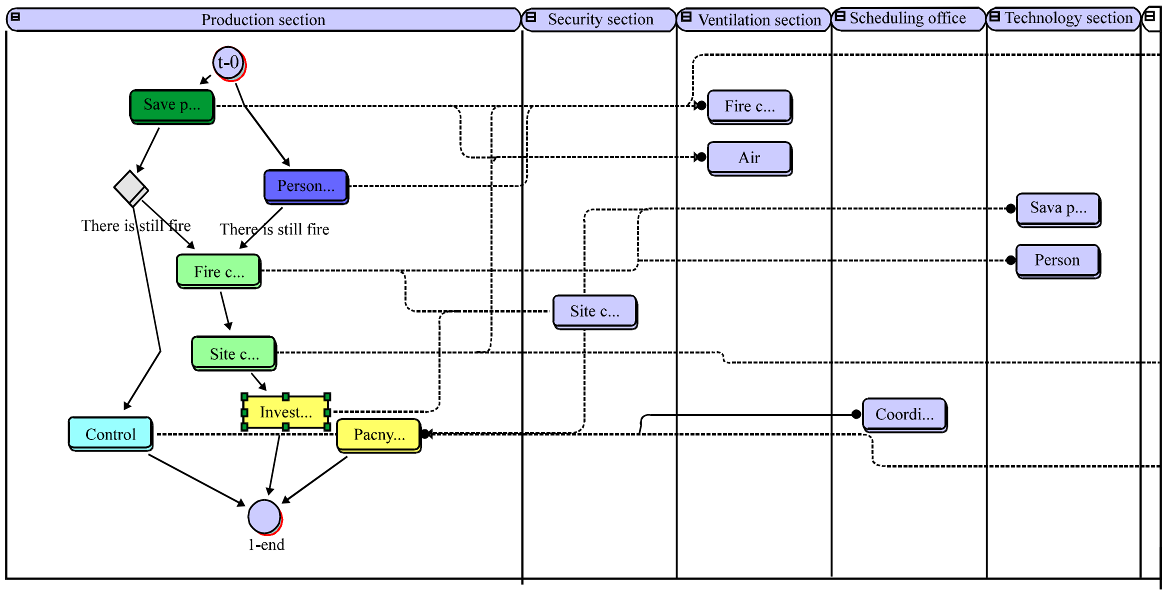

Fig. 18: Organizations monitoring the task-flow in own organization

commanders can edit the task-flow of "Gas Explosion in No.1 Mine", on the basis of the typical task-flow of "Gas Explosion".

After the center commanders creating the task-flow on the basis situation of "Gas Explosion in No.1 Mine", the organizations staff enter the visual environment and monitor the task-flow in their own organizations.

The interface shown in Fig. 18 is the view that people in "Production Section" organization enter the visual environment. People can operate the task nodes in the swimlane of "Production Section", also can view the communication with other organizations.

In the whole rescue process of "Gas Explosion in No.1 Mine" accident, the command center and the organizations keep real-time communication, it is of great value for the rescue work.

CONCLUSIONS

This study introduces the management and monitoring tool in cross-organization ERS, shows the benefit of the tool for monitoring task-flow and, respectively illustrates the different operations for center commanders and organizations staff. According to the task model and typical task-flows, it puts forward a method for recommending the typical task-flow and emergency response tasks. There are some inadequacies in the recommending methods. We will further consider updating the algorithm in calculating the similarity of task-flow models, and the emergency resources scheduling mechanism in task-flow model also deserves a deep study in the future.

ACKNOWLEDGMENTS

This study is supported partly by the NSFC (61170079 and 2009BAG13A02); Sci. and Tech. Development Fund of Shandong Province of China (2010GSF10811); Specialized Research Fund for the Doctoral Program of Higher Education of China (20103718110007), Sci. and Tech. Development Fund of Qingdao (10-3-3-32-nsh), Excellent Young Scientist Foundation of Shandong Province of China (BS2009DX004 and BS2010DX009); the Research Foundation of Shandong Educational Committee (J08LJ77); Natural Science Foundation for Distinguished Young Scholars of Shandong and SDUST (JQ200816 and 2010KYJQ101); Research Project of "SUST Spring Bud" (2010AZZ177 and 2010AZZ069) and Graduate Innovation Fund of SDUST (YCA100339).

REFERENCES

Alfize, A.A., 2002. Metrics evaluation for industrial OO petri nets models. J. Applied Sci., 2: 77-79.

Chrysanthis, P. and K. Rammaritham, 1990. ACTA: A framework for specifying and reasoning about transaction structure and behavior. Proceedings of the ACM SIGMOD International Conference on Management of Data, June 2, 1990, ACM, New York, USA., pp: 72-84.

Cui, T., Q. Zeng and D. Zhang, 2011. Recognition algorithm design and complex analysis for languages of S-nets. Inform. Technol. J., 10: 106-112.

Du, Y.Y. and B.Q. Guo, 2009. Logic petri nets and equivalency. Inform. Technol. J., 8: 95-100.

Fan, Y., 2001. Fundamental of Workflow Management Technology. Tsinghua University Press, Beijing, China, pp: 32-36.

Georgakopoulos, D., M. Hornick and A. Sheth, 1995. An overview of workflow management: From process modeling to workflow automation infrastructures. Distrib. Parallel Datab., 3: 119-153.

Li, J., 2003. Research and Achieve on Visualization Tools of Workflow Modeling. Northwestern University, New York.

Li, P. and Y. Du, 2009. Modeling and design for dynamic workflows based on flexible activities. Inform. Technol. J., 8: 750-756.

Meng, D., Q. Zeng, F. Lu, J. Sun and J. An, 2011. Cross-organization task coordination patterns of urban emergency response systems. Inform. Technol. J., 10: 367-375.

Owaied, H.H., H.A.A.K. Farhan and Y.W. Hudeib, 2011. Framework model for workflow management system. J. Applied Sci., 11: 132-138.

Shahzad, K. and K. Rashid, 2005. A web-enabled architecture of workflow management system for heterogeneous environment. Inform. Technol. J., 4: 356-359.

Sun, J., Q. Zeng, F. Lu, S. Feng and J. An, 2011. Proposal of ontology for resource matchmaking schema in emergency response systems. Proceedings of the 5th International Conference on Knowledge Science, Engineering and Management, December 12-14, 2011, Irvine, CA., USA., pp: 450-461.

WfMC, 1996. Workflow management coalition terminology and glossary (WfMC-TC-1011). Technical Report, Workflow Management Coalition, Brussels.

Zeng, Q., 2008. A construction method for the process expression of petri net based on decomposition. Inform. Technol. J., 7: 420-429.

Zeng, Q., 2011. A polynomial-time decomposition algorithm for petri nets based on indexes of transitions. Inform. Technol. J., 10: 856-862.

Zhan, S., 2008. Research on Emergency Rescue Command System Based on Workflow Technology. Beijing Jiaotong University, Beijing.

Zhu, C.Y., Y.Y. Du and P. Li, 2010. A routing-based dynamic workflow. Inform. Technol. J., 9: 561-568.